New Framework for Semantic Search Engine

Conference Paper · March 2014
DOI: 10.1109/UKSim.2014.114

3 authors, including:

Arooj Fatima
Anglia Ruskin University

Cristina Luca
Anglia Ruskin University

New Framework for Semantic Search Engine

Arooj Fatima, Cristina Luca, George Wilson
Anglia Ruskin University
Cambridge, UK
Arooj.fatima@anglia.ac.uk
Cristina.luca@anglia.ac.uk
George.wilson@.anglia.ac.uk

Abstract— **The semantic web is a technology to save data in a machine-readable format that facilitates machines to intelligently match that data with related data based on meanings. Whilst this approach is being adopted and implemented by some large organisations there is a need for an effective semantic search engine to maximise the full potential of that semantic web. A major difficulty is that the search experience is dependent on a number of elements including a user-friendly interface, a strong query language processor, a result optimiser, result ranking and the use of appropriate data structures to store data. Apart from the technical aspects related to implementation, a strategy to prioritise these elements is needed to optimize and enhance the search experience over the semantic web. The purpose of this work is to investigate some relevant issues on querying the semantic web in a context of semantic search engines, and propose a framework that facilitates an effective search over the semantic web.**

Keywords-component; Semantic Web, Syntax-based Search, Semantic Search, Search Framework

I. INTRODUCTION

There is a large amount of data stored over the Internet that is only useful if accessed as information. To access data from the Internet we need a 'smart' search facility. Search engines are the tools to help users to find data from the huge repository of web pages. To extract data, most of the search engines use syntax-based search or full-text search methods. Full-text searching is a technique whereby a computer program matches terms in a search query with terms embedded within individual documents in a database [4]. An important issue is that "Because full-text searching relies on linguistic matching—matching a word or phrase in a search query with the same word or phrase in a document in the database being searched—it is subject to failure when a variant term exists and is not matched" [5].

Modern syntax based search engines use various techniques (in combination with a full-text search) to compensate for the limitations of a syntax-based search such as page ranking (based on relevancy factor) and content score [15]. To get web pages ranked by the search engines, website developers use a technique called Search Engine Optimisation (SEO). Keywords, meta-tags and micro-formats are the main tools used for SEO. These techniques enhance the factor of user friendliness and increase the chances of more accurate results but these are not the ultimate solution. That is, data searched by a syntax-based search engine has a number of limitations, including high recall with low precision (e.g. thousands of results in response to one or few keywords), low or no recall (when there is no relevant results), and a high sensitivity of results to vocabulary [2].

A semantic web is the optimised solution to these challenges. 'The Semantic Web' can be described as a web of documents linked in such a way so that the data becomes readable and understandable to machines in a meaningful way, hence allowing them to intelligently match related data [6]. One way of viewing this semantic web is that it is a concept of utilising the Internet in such a way that searching the World Wide Web returns results relevant to the meaning of the search query. It is neither exclusively a new way of storing data nor exclusively a new way of querying data, but rather a hybrid approach relevant to existing data whilst supporting new ways of storing data and allowing the utilisation of the semantics of that data to improve the quality of search results.

In recent years, the Resource Description Framework (RDF) has become a popular protocol for storing web-based data with well-defined meanings, and the use of linked data to improve semantic meaning has widened the scope of this protocol. Whilst RDF data is routinely used by many organisations (e.g. gov.uk and bbc.co.uk) its potential to improve semantic searches is now of interest to the database and Internet research communities.

Even if the semantic web is already used in practice, there is a need for a strong semantic search engine to utilise the full potential of that web. Semantic search engines such as Swoogle (swoogle.umbc.edu) and Duckduckgo (duckduckgo.com/) have taken this approach but have a number of limitations especially regarding user experience. This paper investigates some concepts on how the semantic web might be queried in the context of semantic search engines and proposes a framework that facilitates an effective search over the semantic web. Firstly the various factors that influence the search experience over the Internet will be reviewed. Secondly the semantic core technologies necessary to perform a basic search over the Internet will be described - that is RDF and RDF Query Language (SPARQL). Thirdly the academic and social impact of this work is clarified. Finally a proposed framework for a complete search experience for a semantic search engine is presented.

II. RELATED WORK

One part of our proposed system is quite similar to that described in [17] whose author's proposed a framework for SPARQL query processing, optimisation (by re-writing queries), execution and evaluation. However, their framework relates to the core query elements of RDF and SPARQL whilst we explore all the elements that affect the querying process over the semantic web in the context of a semantic search engine. Another closely related approach proposed by [10] is a move towards ontology-based information retrieval where they discuss effective ranking and semantic search techniques. However, their work is a generic approach to semantic information retrieval while the scope of our proposed framework explores all aspects affecting the usability and reliability of a semantic search engine.

III. CHALLENGES FOR A SEMANTIC WEB SEARCH ENGINE

A semantic web search engine should be able to search data over the Internet with maximum precision and accuracy and be able to link related data. Such a search engine should consider the following criteria: user experience, efficiency, page ranking, scalability, and cost effectiveness.

A. User Experience

A friendly user interface is the most important feature that will enhance the user experience. Search engines such as Google, Bing and Yahoo have all been through a number of improvements in order to provide end-users with the best possible user experience. Even if the results are incomplete or sometimes not accurate due to syntax-only based search algorithms, end-users still adhere to these search engines for the sake of a good perceived user experience. Improvements to the end-user interface of a semantic query search engine needs substantial development so that poor input representation of a query will automatically suggest corrections for spelling mistakes and poor grammar, and of course find the best matched results with a high accuracy.

B. Efficiency

An efficient (i.e. fast in this context) search engine's performance depends upon the size of data to be matched, the request time to the web servers and the associated response time. For a semantic web query the execution time also depends on factors such as delays caused by looking up Uniform Resource Locators [8], efficient support for indexing large-scale data [7], and dealing with query termination problem [9].

C. Page Ranking

The main idea of the Semantic Web is to retrieve the most relevant and accurate results in response to a query. Page ranking is a way of grading web pages so that the web pages with the highest ranking are presented at the top of a list of search results [21]. This is a challenging task given that there are "more than 11.3 billion web pages in the World Wide Web" at the time of writing [15], and a single user query on a search engine may return millions of results. It is therefore crucial that the search engine can sort and rank the retrieved documents effectively in order of either relevancy or authenticity. There are a number of techniques used by search engines to rank page results. Page ranking methods can arrange results in order of relevance, importance and content score [15]. The syntax-based search engines use various algorithms to calculate relevancy of a web page to the search terms and arrange the results sorted with highest relevancy at the top.

D. Scalability

Scalability within the context of a search engine is the ability of a system to handle a rapidly growing amount of data. Relational database management systems have repeatedly shown that they are very efficient, scalable and successful [4] and indeed their success is due to the structure of relational data. However scalability for data in a semantic web presents additional challenges because of the openness of the RDF protocol.

E. Cost Effectiveness

A good search system must provide a solution that is cost effective. Due to the openness of RDF, the queries can be quite expensive while dealing large data sets. For efficient data retrieval, conventional search engines use indexing techniques at the cost of additional storage. As a semantic search engine processes an open structure like RDF, complex queries can be very expensive to process. Only a few solutions have been proposed to solve this issue (e.g. query caching). Some efforts have been made to introduce cost effective search algorithms such as the SPARQL and RDF incorporation of greedy search techniques [7].

IV. ANALYSIS OF SEMANTIC WEB TECHNOLOGIES

Two principal aspects of a semantic search are relevant here; namely data storage (via RDF) and data extraction (via a Query Language).

A. Data Language

There are various techniques used to store data, including as discrete files, structured files as Extensible Markup Language (XML), and use of relational databases. While the semantic web has an architectural scope that varies from the standard web architecture, it optimally needs a different database structure. The core of the semantic web is built on the RDF model [18]. RDF is both a language and a structure, rather like XML but is far advanced and has a totally different functionality. RDF describes a resource as a triple (subject, predicate, object). An object in one predicate can be the subject of another predicate implementing the concept of linked data in a schema free structure [24]. The concept of linked data makes the scope of the semantic web far broader than the standard web, but much research is needed as to how the querying of linked data can be efficiently optimised [3].

B. *Query Language*

In spite of certain limitations, the Semantic Web is already an actuality and there are already a number of published RDF resources available over the Internet. To utilise the available RDF data we need an effective search facility. Since RDF utilises a different structure to store data than the conventional methods of data storage, it requires a query language that is compatible to its structure and yet efficiently extracts data. There are a number of query languages that have been developed to fulfil this requirement (e.g. rdfDBL, SquishQL, Jena-RQL, RDQL, SerQL, MashQL and SPARQL) but most have not persisted because of their inherent limitations. Whilst SPARQL is the existing standard for RDF it too has certain limitations including restricted navigational functionalities, complexity of evaluation and a lack of consideration for potentially infinite numbers of RDF interpretations [3, 22]. These limitations highlight the need for a query model that is applicable to linked data across what is essentially a highly distributed system; there is a need for a semantic search engine to be able to answer queries over virtual SPARQL views whilst enforcing security policies and integrating data from different sources [14]. The SPARQL working group and related research communities are actively involved in activities to explore and update SPARQL features. A number of extensions have already been introduced to eliminate SPARQL limitations including cpSPARQL, nSPARQL and pSPARQL [1].

V. IMPLICATIONS FOR A SEMANTIC SEARCH ENGINE

Given the capacity of existing semantic web technologies, it is clear that the semantic web is finally coming into focus via a number of implementations developed by some large organisations and despite the current limitations of the semantic web technologies (RDF and SPARQL in particular). The semantic web is therefore in need of some effective search engines that express the power and strength of being semantic.

Whilst the semantic web is based on new technologies with different data structures, it requires new tools to facilitate the semantic search queries. The responsibility of a semantic search engine is not limited to solely extract data from RDF repositories alone, given there is a very small percentage of RDF (or embedded RDF) documents published on the Internet. The semantic search engine needs to deal with both RDF and non-RDF data whilst handling all the challenges faced by a standard search engine.

Internet users are used to having a good user experience with existing search engines (e.g. customisation, saving search history, auto-correct and auto-suggest options) even though that does not necessarily involve the best-matched results. Though the main focus of a semantic search is to retrieve best matched results with highest relevancy and accuracy it is imperative that the user-friendly features of conventional search engines are retained in any new semantic search engine implementation. This paper proposes a framework that takes care of all aspects necessary for a semantic search engine.

VI. A FRAMEWORK FOR A SEMANTIC SEARCH ENGINE

Our proposed framework for a semantic search engine that fulfils the requirements of returning best-matched results whilst retaining user-friendliness is shown in Figure 1.1.

A. *User Interface*

Usability is one of the main challenges for a semantic search engine. A query engine can only work effectively if it actually knows what to search for.

In a conventional programming environment the developers will write queries knowing the search criteria, data structure (e.g. relational database) and the query language. However a truly optimised semantic search requires a search engine that can itself write queries to be implemented by the query engine. So a query language alone is not enough to benefit a search system but also depends upon the base data structure it works upon and the top-level system interacting with it. This functionality must not compromise the qualities of a conventional end-user interface and should include user-friendly features such as auto-suggest and auto-correct functionality [13] and an ontological classification of words possibly by query-tagging [19].

In the present work the authors propose an interface that provides a facility to tag search terms (keywords) with related ontology. At a basic search level, the interface smartly re-phrases the keywords based on its own understanding and suggests some query sentences to the end-user. If the interface is not able to understand the query, it provides the option of a further advanced search where the user can tag each search term from a list of related ontologies available for that term. For example for a search on "java and apple" the user can tag "java" as {language or place} and "apple" as {fruit or device}.

B. *Query Optimiser*

A major challenge for a semantic search engine is to translate a natural language query to a search query that is understandable by the query engine. The processing of the semantic web query language is different from other query languages because it does not work on pre-written queries by program developers that simply match the search criteria with different pre-defined data fields (as in relational database) or nodes (like XML). Instead it dynamically writes queries using various algorithms at initial query runtime.

To write runtime queries, the user keywords should be mapped to their relative ontologies. The same approach has been explored by [11] where they modeled some test inputs entered by the user with related ontology although it is not clear that how the input is supposed to be entered. End-users are rarely specific while writing search keywords and always expect the search engine to understand what they are looking for. We propose a system in this paper that takes care of this issue by introducing the 'Query Optimiser'.

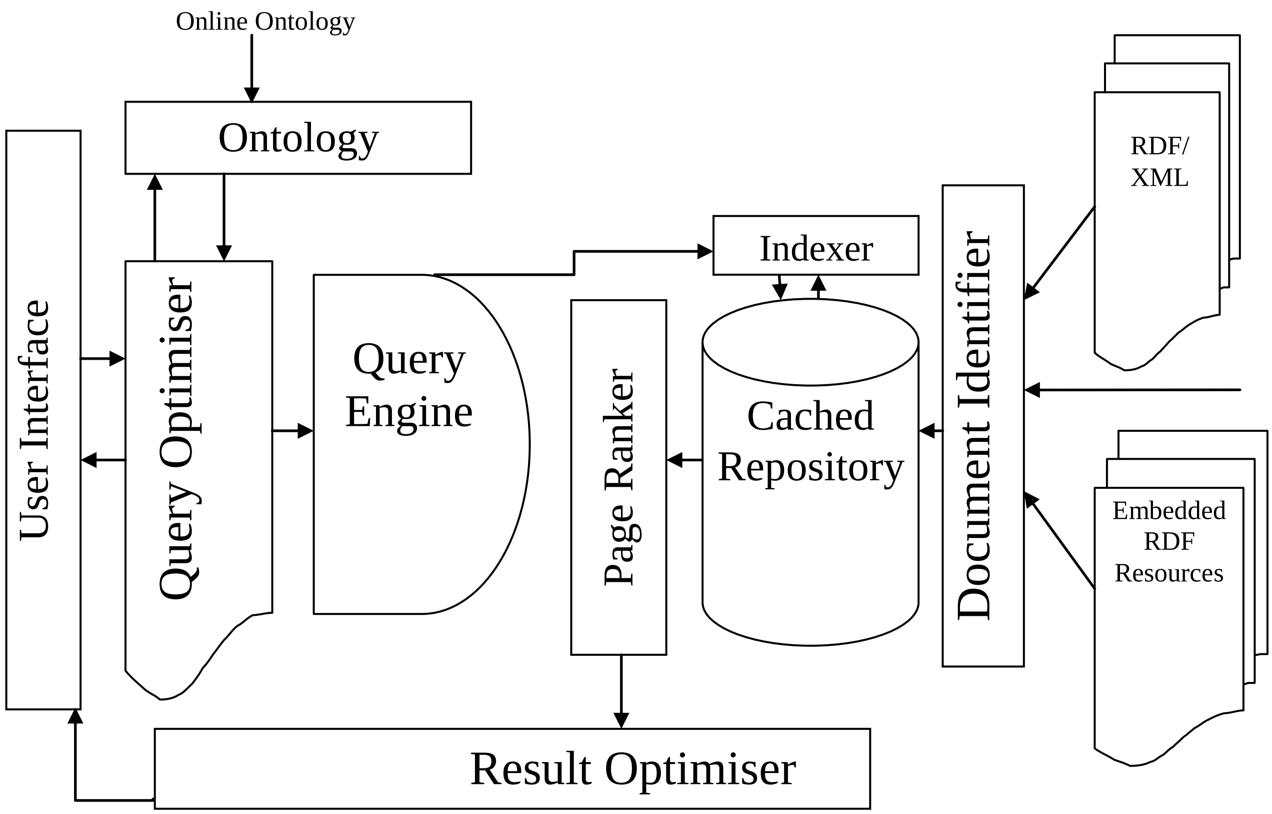

Figure 1. Our Framework for Semantic Search Engine

The query optimiser scans the keywords entered by the end user and checks if it can understand the query. If the entered search terms do not make complete sense to the optimiser it checks for grammatical errors first and suggests some options to the user if required. It also supports advance searching (as discussed earlier) where the user can clarify various terms by relating them to pre-defined vocabularies/ontologies using an 'ontology processor'.

C. *Ontology Processor*

The ontology processor plays a vital role for query optimisation. It will help the query optimiser to suggest tags to query keywords where tags will have associated ontologies.

The retrieving of ontologies from an online store or library (every time a user enters search input) in order to tag a word is a lengthy process that will have a cost in terms of efficiency. Unavailability or slow response of an online ontology server can also limit the functionality of the search engine. To overcome this issue, the proposed ontology processor consists of two parts:

1) *Ontology Cache*

One part of the ontology processor will be a service that will include a 'cron job' (that is, a scheduled job on a server), which regularly updates an ontology list from an ontology store and retains a cached copy of the online ontologies.

2) *API Service*

The second part of the ontology processor will work like an API (Application Program Interface) that will receive key words from the Query Optimiser and return matching tags with associated ontologies as a response.

D. *Query Engine*

The query engine is the core part of the proposed framework. Of the different semantic query engines that have been proposed SPARQL is the one existing standard that supports the W3C standard and is continuously being updated by a SPARQL working group. Whilst the feasibility of SPARQL is questionable in certain aspects it is outside the scope of this work to propose which query language should be used. Most of the semantic search systems including SPARQL support formal queries where the user needs to know the SPARQL language and schema of RDF [16]. A user of a semantic search engine writes a search query in a natural language without knowing the technical details. So a facility is needed to translate natural language to the query language syntax. The query optimiser passes search/key words with related tags to the query engine. The query engine consists of the following parts:

1) Query Formatter

It retrieves tagged keywords from the query optimiser and formulates a query using namespaces (for ontologies) from tags and search terms.

2) Query Validator

It checks the syntax and validates the query formulated by query formatter. If the query passes validation, it is passed on to the query processor.

3) Query Processor

It runs search algorithms on the cached repositories (which will be cached for efficiency). The query engine requests the indexer to see if some pages have been already indexed against the search query.

E. Indexer

For a graph as large as the Internet millions of iterations may be necessary to search for a matching result and this raises the problem of query termination, processing time and cost of processing. To deal with this issue search engines normally use a back-end activity where search-bots and crawlers navigate the Internet and retain an updated cached copy of the web pages. To enhance the efficiency of information retrieval, the conventional search engines deploy ‘indexing’ using their defined algorithms.

Indexing is a technique to speed up the search tasks. There are different techniques used for indexing (e.g. individual presentation, combined presentation and quad-combined presentation) and research shows that the combined presentation technique is more useful [9]. The quality of query answers depends on the source selection (i.e. selecting potentially relevant data) [23] but the quality of search tool depends upon the speed of informal retrieval. Indexing speeds up the process though at the cost of more storage space.

For efficient data retrieval, the proposed framework uses the conventional approach of caching documents. The ‘Indexer’ stores information about cached documents (on the server). Whilst documents are indexed, the query engine does not need to search through all available documents while running a search query. Like a standard indexing tool the proposed ‘Indexer’ indexes significant keywords and also stores information about matched Uniform Resource Identifiers (URIs), base Uniform Resource Locators (URLs) and linked namespace.

F. Page Ranker

Semantic data page ranking is a challenging task as the search engine sees the Internet as a giant graph. The page ranker in our framework is based upon the idea presented by [12] that proposes an approach called an “ontology concept” that calculates the ranking of search results using frequency of the ontology used. However a higher frequency (of ontology usage) does not mean higher relevancy. To accommodate this the ‘Page Ranker’ will build a record of trusted web pages against distinct search terms. The trust level will be calculated using various techniques such as hit counts, bounce rate and listing authenticated domains. In addition to this approach, the Page Ranker will have smart functionality to arrange pages using various customised criteria (e.g. order of relevance, order of occurrence, order of published date, user location). It also uses built in algorithms to rank non-RDF pages based on best possible match and frequency of search terms used.

G. Document Identifier

Considering that there is a huge amount of data over the Internet that is not in RDF format, alternative strategies are needed to search such data resources. RDF can be embedded into languages like PHP and HTML but currently there is no well defined way to extract the data. Conversely it is not easy to migrate all non-RDF data over the Internet into RDF, and convincing the data owners to migrate their data into RDF format is also a big challenge (owners may not understand the benefits of the semantic web and the benefits may not be evident until the semantic web is realised – a circular problem). Therefore to make the semantic web fully functional in the short term, both RDF and non-RDF data must be accessible until such a time that most of the internet data is converted to RDF, or until some appropriate conversion techniques are found.

As a starting point for non-RDF data, the framework model of this work proposes the inclusion of additional metadata in the HTML header that define namespaces for the main ontologies that are well suited to the data used in that web page. While website owners already pay a lot of money for SEO, it is not at all difficult to add some namespaces with other data. For SEO a meta-tag is defined in HTML with a syntax:

<meta name="meta_data_name"
content="description_of_data">

and there can be as many meta-tags defined as required. We propose to use the same meta-tags to define namespaces, e.g.

<meta name="namespace" content = "uri_of_namespace : comma_separated_keywords" >

The comma separated keywords will be related to the namespace defined before the colon (:). This way as many namespaces as required can be defined with their related keywords. The document identifier filters documents grabbed and cached from the Internet and processes each document accordingly. It will index and tag all documents as RDF, embedded RDF, meta-RDF and non-categorised HTML pages. This way no data will remain unlisted though the ranking will prioritise the data with associated namespaces.

H. Result Optimiser

The Result Optimiser optimises the results to be sent to the user interface e.g. grouping and ordering results with user preferences (if any) or sorting by user location from where that person is logged in. The result optimiser also considers controlling duplicate results if any are found. For eliminating duplicate query results the technique used by [20] is proposed for this work where authors combine the page hash and page size (although this approach still needs to be tested at a large scale).

VII. CONCLUSION

The semantic web is a flourishing technology and its implementations are rapidly growing. The feasibility of the core idea of the semantic web is gaining acceptance and this acceptance is opening many research areas in the effective use of semantic technologies. One of the questions is how to effectively implement a semantic web for a semantic search engine. There are certain elements involved in querying semantic web data over the Internet in the context of a search engine. These elements include a user-friendly interface, a strong query language processor, a result optimiser, result ranking and indexing. A query engine (in the context of a search engine over the Internet) has functional and non-functional dependencies on these elements. The ability to use non-RDF data is another challenge that developers of semantic search engines must overcome. The proposed framework for a semantic search engine presented in this paper aims to address these issues. Future work will develop individual elements of the framework prior to their integration.

REFERENCES

[1] F. Alkhateeb, J.Euzenat, "Answering SPARQL queries modulo RDF Schema with paths". Research Report, Yermouk University, Jordan, 2013.

[2] G. Antoniou, F. V. Harmelen, "A Semantic Web Primer", 2nd Edition. London: MIT Press, 2008.

[3] M. Arenas and J. Perez, "Querying the Semantic Web Data with SPARQL", Proc. of the thirtieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems (PODS 11), ACM, New York, NY, USA, 2011 pp. 305-316, doi:10.1145/1989284.1989312.

[4] J. Beal, "Weaknesses of Full text search", The Journal of Academic Librarianship, vol. 34, Number 5, 2008, pp. 438-444. doi: http://dx.doi.org/10.1016/j.acalib.2008.06.007.

[5] J. Beal, "Geographical research and the problem of variant place names in digitized books and other full-text resources" Library Collections, Acquisitions, and Technical Services, vol. 34, Issues 2–3, 2010, pp. 74-82. doi: http://dx.doi.org/10.1016/j.lcats.2010.05.002.

[6] T. Berners-Lee, "Weaving the Web", pp. 2-5. The Original Design and Ultimate Destiny of the World Wide Web by its Inventor. Harper :San Francisco, 1999.

[7] L. Chang, W. Haofen, Y. Yong and X. Linhao, "Towards Efficient SPARQL Query Processing on RDF Data", Tsinghua Science & Technology, vol. 15, issue 6, Dec. 2010, pp. 613-622. doi:10.1016/S1007-0214(10)70108-5.

[8] O. Harting and F. Huber, F., "A main memory index structure to query linked data". Proc. 4th Linked Data On the Web (LDOW11), March 2011.

[9] O. Harting and J. Freytag, J "Foundations of traversal based query execution over linked data", Proc. 23rd ACM conference on Hypertext and social media (HT '12). ACM, New York, NY, USA, 2012, pp. 43-52, doi=10.1145/2309996.2310005.

[10] S. Kara, Ö. Alan, O. Sabuncu, S. Akpınar, N. K. Cicekli and F. N. Alpaslan, "An ontology-based retrieval system using semantic indexing", Information Systems, vol. 37, issue 4, June 2012, pp. 294-305. doi:10.1016/j.is.2011.09.004.

[11] M. Shekhar and RA. K. Saravanaguru, "A case study on semantic web search using ontology modeling", International journal of engineering and technology, vol. 5, no. 3, 2013, pp. 2342-2348, ISSN 0975-4024.

[12] A. Kayed, E. El-Qawasmeh and Z. Qawaqneh, "Ranking web sites using domain ontology concepts", Information & Management, vol. 47, issues 7–8, Dec. 2010, pp. 350-355. doi: http://dx.doi.org/10.1016/j.im.2010.08.002.

[13] A. Latif, M.T. Afzal, P. Hoefler, A.U. Saeed, and K. Tochtermann, "Turning keywords into URIs: simplified user interfaces for exploring linked data", Proc. of the 2nd International Conference on Interaction Sciences: Information Technology, Culture and Human (ICIS '09). ACM, New York, NY, USA, 2009 pp. 76-81. Doi: http://doi.acm.org/10.1145/1655925.1655939.

[14] W. Le, S. Duan, A. Kemenstsietsidis, F. Li and M. Wang, "Rewriting queries on SPARQL views", Proc. of the 20th international conference on World wide web (WWW '11). ACM, New York, NY, USA, 2011, pp. 655-664, doi:http://doi.acm.org/10.1145/1963405.1963497

[15] M. P. Selvan, C. A. Sekar and P. A. Dharshini, "Survey on Web Page Ranking Algorithms", International Journal of Computer Applications, vol. 41, no.19, Published by Foundation of Computer Science, March 2012, doi: 10.5120/5646-7764.

[16] H. Li and Y. Wang, "Ranked keyword query on Semantic Web data", Proc. Fuzzy Systems and Knowledge Discovery (FSKD), 2010 Seventh International Conference on , vol.5, Aug. 2010, pp.2285-2289, doi: 10.1109/FSKD.2010.5569309.

[17] S. K. Malik, S. Rizvi, "A Framework for SPARQL Query Processing, Optimization and Execution with Illustrations". International Journal of Computer Information Systems and Industrial Management Applications. Vol. 4, 2012, pp. 208 -218, ISSN 2150-7988.

[18] F. Manola and E. Miller, "RDF Primer", W3C Recommendation, Feb. 2004.

[19] M. Manshadi and X. Li, "Semantic tagging of web search queries", Proc. of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: vol. 2, 2009, pp. 861- 869, ISBN: 978-1-932432-46-6.

[20] O. Naseer, A. Naseer, A. A. Khan H. and Naseer, "Using Page Size for controlling duplicate Query Results in Semantic Web". International Journal of Web and Semantics (IJWesT) Vol.4 No.2 April 2013.

[21] G. P. Schneider and J. Evans, "New perspectives on the Internet". Ohio : South-Western, 2012

[22] F. Picalausa and S. Vansummeren, "What are real SPARQL queries like?", Proc. of the International Workshop on Semantic Web Information Management (SWIM '11). ACM, Article 7, 2011, doi:10.1145/1992999.1999306.

[23] A. Qasem, D.A. Dimitrov, J. Heflin, "Efficient Selection and Integration of Data Sources for Answering Semantic Web Queries", Proc. of the International Conference on Semantic Computing, IEEE, Aug. 2008, pp.245-252, doi:10.1109/ICSC.2008.31.

[24] S. Sakr and G. Al-Nymat, "Relational processing of RDF queries: a survey", ACM SIGMOD Record, vol. 38, issue 4, Dec. 2009, pp. 23-28, doi:10.1145/1815948.1815953